Computational Finance - Lab Assignment 1

Contact: k.chatziandreou@uva.nl // Submit until 23.02.2026

1 Financial Data Science: Realized Volatility [6p]

In this first assignment, we consider the problem of estimating the parameters μ and σ from market data under the stock price model (working under the physical measure $\mathbb{P}$):

$$\frac{dS_t}{S_t} = \mu dt + \sigma dB_t. \tag{1}$$

Tasks: First, read AppendixA carefully. Then follow the steps:

1. **Step 1: Install Python Libraries to Retrieve Financial Data**
   In a command prompt or terminal, run:

   ```
   pip install skfolio
   ```

   `skfolio` is a Python library for portfolio optimization built on top of scikit-learn. We will require this library to load the 3 months ATM implied volatility of the 20 assets from the SP500 dataset. experiment with different ones.

2. **Step 2: Explore the Stooq API via `pandas_datareader`**
   Select a ticker of your choice and specify a start and end date (ideally covering a long historical period) to download daily price data and then plot estimators for your mean and realized variance. For example:

```python
import datetime as dt
import pandas as pd
from pandas_datareader import data as pdr

def download_prices_stooq(ticker: str, start_date: str, end_date: str) -> pd.DataFrame:
    """
    Download daily OHLCV from Stooq using pandas_datareader.
    For US stocks, Stooq often uses the format 'AAPL.US'.
    """
    start = pd.to_datetime(start_date)
    end = pd.to_datetime(end_date)

    candidates = [
        ticker,
        ticker.upper(),
        f"{ticker.upper()}.US",
        f"{ticker.upper()}.US"
    ]

    last_err = None
    for t in candidates:
        try:
```

```
            df = pdr.DataReader(t, "stooq", start, end)
            if df is not None and not df.empty:
                # Stooq returns newest->oldest; sort to oldest->newest
                df = df.sort_index()
                df.index = pd.to_datetime(df.index)
                return df
        except Exception as e:
            last_err = e

    raise RuntimeError(
        f"Failed to download data for {ticker} from Stooq. "
        f"Tried: {candidates}. Last error: {last_err}"
    )

# ----------------------------
# Parameters
# ----------------------------
ticker = "AAPL"
start_date = "2010-01-01"
end_date = dt.date.today().strftime("%Y-%m-%d")

# ----------------------------
# Download data (Stooq)
# ----------------------------
data = download_prices_stooq(ticker, start_date, end_date)

# Use Close (Stooq doesn't always provide Adj Close)
stock = data["Close"].astype(float)
```

3. Step 3: Historical Estimators

- Select a ticker and a start and end date (ideally a long time window) to plot estimators for the mean and realized variance using the adjusted closing price.
- Compute the classical historical mean and volatility estimators (as referenced in equations (8) and (13) in your notes).
- Implement one alternative volatility estimator of your choice (e.g., Parkinson or Garman–Klass) and compare it with the classical volatility estimator. For example, the Parkinson estimator is given by

$$\sigma_{\text{Parkinson}} = \sqrt{\frac{1}{4\ln 2}\sum_{t=1}^{T}\left[\ln\left(\frac{h_t}{l_t}\right)\right]^2}, \tag{2}$$

where T is the number of days in the sample, h_t is the high price, and l_t is the low price on day t.

Similarly, the Garman–Klass estimator is defined as

$$\sigma_{\text{Garman–Klass}} = \sqrt{\frac{1}{2T}\sum_{t=1}^{T}\left[\ln\left(\frac{h_t}{l_t}\right)\right]^2 - \frac{2\ln 2 - 1}{T}\sum_{t=1}^{T}\left[\ln\left(\frac{c_t}{o_t}\right)\right]^2}, \tag{3}$$

where o_t is the open price and c_t is the close price on day t.

- Compute a rolling window estimator using a 30-day window. Plot the results for all estimators and explain your findings.
- Finally, select one of the S&P 500 stocks and compare the realized variance estimators with the time series of implied volatilities for that ticker. To retrieve implied volatility data, you can use the `skfolio` package. For example:

```
import pandas as pd
from skfolio.datasets import load_sp500_dataset, load_sp500_implied_vol_dataset
from skfolio.preprocessing import prices_to_returns

prices = load_sp500_dataset()
implied_vol = load_sp500_implied_vol_dataset()
X = prices_to_returns(prices)
X = X.loc["2010":]
implied_vol.tail()

```

4. **Step 4: Implied Volatility Data and VIX Estimation**

(a) Download the SPX and VIX quoted data for the specified time window:

```
import numpy as np
import pandas as pd
import datetime
import math
from pandas_datareader import data as pdr

def fetch_spx_close(start_date, end_date):
    """
    Fetch an SPX-like daily close series.
    Primary: FRED SP500
    Fallback: Stooq ^SPX
    Returns a DataFrame with a 'Close' column and DatetimeIndex.
    """
    start = pd.to_datetime(start_date)
    end = pd.to_datetime(end_date)

    # Try FRED first
    try:
        df = pdr.DataReader("SP500", "fred", start, end)
        df = df.rename(columns={"SP500": "Close"}).dropna()
        df.index = pd.to_datetime(df.index)
        return df
    except Exception:
        pass

    # Fallback: Stooq
    df = pdr.DataReader("^SPX", "stooq", start, end)
    df = df.sort_index()
    df.index = pd.to_datetime(df.index)
    return df[["Close"]].dropna()

def fetch_vix_close(start_date, end_date):
    """
    Fetch daily VIX close series.
    Primary: FRED VIXCLS
```

```
    Fallback: Stooq VI.F (VIX)
    Returns a DataFrame with a 'Close' column and DatetimeIndex.
    """
    start = pd.to_datetime(start_date)
    end = pd.to_datetime(end_date)

    # Try FRED first
    try:
        df = pdr.DataReader("VIXCLS", "fred", start, end)
        df = df.rename(columns={"VIXCLS": "Close"}).dropna()
        df.index = pd.to_datetime(df.index)
        return df
    except Exception:
        pass

    # Fallback: Stooq (try common symbols)
    for sym in ["VI.F", "vi.f", "^VIX", "^vix"]:
        try:
            df = pdr.DataReader(sym, "stooq", start, end)
            df = df.sort_index()
            df.index = pd.to_datetime(df.index)
            if "Close" in df.columns and not df.empty:
                return df[["Close"]].dropna()
        except Exception:
            continue

    raise ValueError("No VIX data found from FRED or Stooq.")

spx_symbol = "^SPX"
today = pd.to_datetime("2025-03-05")  # Keep this fixed in your implementation
end_date = today
start_date = end_date - datetime.timedelta(days=365)

spx_data = fetch_spx_close(start_date, end_date)
if spx_data.empty:
    raise ValueError("No SPX data found from FRED/Stooq for this window.")

lastBusDay = spx_data.index[-1]
S0 = float(spx_data["Close"].iloc[-1])  # spot/closing price

# Fetch VIX in a short window after lastBusDay (first available close)
vix_data = fetch_vix_close(lastBusDay, lastBusDay + datetime.timedelta(days=30))
if vix_data.empty:
    raise ValueError("No VIX data found from FRED/Stooq for this window.")

vix_market = float(vix_data["Close"].iloc[0])

# Fixed inputs (keep these fixed in your implementation)
T = 30/365.0
r = 0.02
F0 = S0 * math.exp(r * T)  # forward approximation

print("Last Bus Day:", lastBusDay)
print("S0:", S0)
print("VIX market close:", vix_market)
print("F0:", F0)

```

(b) To determine the available expiration dates for a particular ticker, use an options data interface. For example, one option is `yahooquery`:

```
import pandas as pd
from yahooquery import Ticker

spx_ticker = Ticker("^SPX")

# Option chain table (may be empty depending on data availability)
oc = spx_ticker.option_chain

if oc is None or (isinstance(oc, pd.DataFrame) and oc.empty):
    print("No option chain returned. If needed, use the CSVs provided on Canvas.")
else:
    # Available expirations are contained in the MultiIndex level "expiration"
    expirations = oc.index.get_level_values("expiration").unique()
    expirations = sorted(pd.to_datetime(expirations).strftime("%Y-%m-%d").tolist())
    print("Available expirations:", expirations)
    # Suppose the next expiration is "2025-04-03" expiry_date = "2025-04-03"  # Fixed to
     approximate a 30-day horizon as per CBOE

# If you cannot download the chain, use the CSVs on Canvas:
# Call_option_data_2025-04-03.csv and Put_option_data_2025-04-03.csv

```

5. Compute the estimated VIX using the estimator VIX_t (referenced as equation (18) in your notes) and compare it with the CBOE-quoted VIX.
6. Plot the historical estimated realized variances from **Step 3** alongside the VIX time series. Perform statistical analyses (such as correlation or cointegration tests) to assess the relationship between the time series.
7. Run regression analyses between SPX returns and the VIX index, as well as between SPX returns and the historical realized variance estimator. Discuss your observations.
   Hint: The variations of the stock index are typically negatively correlated with variations of the VIX index, whereas the correlation with historical volatility variations may differ.

2 Hedging: Volatility Mismatch [8p]

Consider a short position in a European call option on a non-dividend paying stock with a maturity of one year and strike $K = 99$ EUR. Let the one-year risk-free interest rate be 6% and the current stock price be 100 EUR. Furthermore, assume that the volatility is 20%.

Use the Euler method to perform a hedging simulation.

Tasks:

1. **Matching Volatility:** Conduct an experiment where the volatility in the stock price process matches the volatility used in the delta computation (i.e., both set to 20%). Vary the frequency of hedge adjustments (from daily to weekly) and explain the results.

2. **Mismatched Volatility:** Perform numerical experiments where the volatility in the stock price process does not match the volatility used in the delta valuation. Run computational experiments for various levels of volatility and discuss the outcomes.

3. **Pricing and Hedging with Implied Volatility:** Assume that for an underlying with spot price S_t, a vanilla option with value function

$$C(t, S; \sigma_{\text{imp}}) := C_t,$$

is priced under the risk-neutral measure $\mathbb{Q}$ by solving the Black–Scholes–Merton PDE with implied volatility σ_{imp}:

$$rC_t - rS_t \frac{\partial C_t}{\partial S_t} - \frac{\partial C_t}{\partial t} - \frac{1}{2}\sigma_{\text{imp}}^2 S_t^2 \frac{\partial^2 C_t}{\partial S_t^2} = 0, \tag{4}$$

together with appropriate terminal conditions. The option delta is given by

$$\Delta_t = \frac{\partial C_t}{\partial S_t},$$

and is computed by solving (4). We also assume that the volatility σ_{imp} is implied from a market quote for $C(0, S; \sigma_{\text{imp}})$ and remains fixed until maturity T. This means the option is priced and delta-hedged at the level of the implied volatility, thereby avoiding exposure to directional changes in the underlying when hedging with a different volatility.

In practice, delta-hedging is performed in discrete time (typically daily), and real securities do not follow perfect log-normal diffusive processes. Thus, assume that the dynamics of the underlying price S_t under the objective (real-world) measure $\mathbb{P}$ are given by

$$\frac{dS_t}{S_t} = r\,dt + \sigma_t\,dW_t^{\mathbb{P}}, \tag{5}$$

where σ_t denotes the instantaneous realized volatility. Model parameters in this SDE are estimated for trading and risk management purposes, and the expected profit and loss (P&L) and its variance are computed under $\mathbb{P}$.

Show that in order for the final P&L to vanish on average, the implied volatility σ_{imp} used for pricing and risk management must, on average, match the future realized volatility σ_t when weighted by the option's dollar gamma over its life. That is, one must have

$$\mathbb{E}\left[\int_0^T e^{-rt}\frac{1}{2}S_t^2\,\frac{\partial^2 C_{\sigma_{\text{imp}}}}{\partial S_t^2}\left(\sigma_{\text{imp}}^2 - \sigma_t^2\right)dt\right] = 0, \tag{6}$$

where σ_t is the instantaneous realized volatility.

3 Hedging with Real SPX Option Data

In this assignment you will implement and evaluate **delta hedging strategies** for SPX index **European call options** using real market data. The assignment is **competitive**: you must report a standardized set of calibration and hedging metrics so that results are comparable across students.

Learning objectives

By completing this lab you will:

- implement one-step (one-day) delta hedging strategies using real option data,
- compare hedging effectiveness using **SSE** and the **Gain** metric.
- Study effects of model's misspecification on hedging performance.

Dataset

The dataset consists of SPX call option quotes over the period **2023-02-01 to 2023-02-28** (WRDS export). For each trading date t and each option contract (strike K, expiry T), the dataset contains at least

(`date`, `exdate`, `symbol`, `strike_price`, `best_bid`, `best_offer`, `impl_volatility`, `delta`, `cp_flag`).

You must use `cp_flag = 'C'` (calls only).

Underlying SPX close series. Let S_t denote the SPX close on date t. If S_t is not included in the WRDS export, you must source it externally and merge it by date. **Important:** since the evaluation uses one-day changes $\Delta S_t = S_{t+1} - S_t$, you must obtain *at least one additional trading day after 2023-02-28.*

Notation

Trading days and one-day changes. Let t index trading days. For any end-of-day quantity X_t, define

$$\Delta X_t := X_{t+1} - X_t.$$

Midquote. Define the call option midquote V_t (in index points) by

$$V_t := \frac{\text{bid}_t + \text{ask}_t}{2} \quad \text{(use best bid/offer if available).}$$

Maturity. Define the days-to-expiration and year fraction:

$$D_t := (T - t) \text{ in days}, \qquad \tau_t := \frac{D_t}{365}.$$

(If you use a different day-count convention, you must apply it everywhere consistently.)

Underlying. Let S_t denote the SPX close, with one-day change $\Delta S_t := S_{t+1} - S_t$.

Delta-hedged residual (one-step). Given a delta hedge process δ_t, define the hedging residual

$$\varepsilon_t(\delta) := \Delta V_t - \delta_t \, \Delta S_t.$$

SSE and Gain. For a set of observations $\mathcal{I}$, define

$$\mathrm{SSE}_{\mathcal{I}}(\delta) := \sum_{t \in \mathcal{I}} \varepsilon_t(\delta)^2.$$

Given two delta hedges $\delta^{(A)}$ and $\delta^{(B)}$, define the Gain

$$\mathrm{Gain}(A \text{ vs } B) := 1 - \frac{\mathrm{SSE}(\delta^{(A)})}{\mathrm{SSE}(\delta^{(B)})},$$

where both SSEs are computed over the same observation set.

Important: Standardization for competition. You must follow the filtering and bucketing rules in Step 6 exactly. Your competitive score will be based on SSE and Gain computed after these rules.

Tasks: Follow the steps below.

1. **Step 1: Data loading and cleaning (reproducibility)**
   Load the option panel and apply the following mandatory preprocessing:
   - Keep calls only.
   - Convert dates to a consistent date format.
   - Rescale strikes if needed so that strikes are in index-point units (In our case just devide by 1000 for SPX options).
   - Construct the midquote V_t.
   - Remove observations with missing or invalid implied volatility.

   Required reporting:
   - row counts before/after filtering to calls,
   - number of unique trading dates and unique expiries,
   - missing rates of $(V_t,\ \sigma_{\mathrm{mkt}},\ \delta)$,
   - the list of final columns you keep for the project.
2. **Step 2: Contract time series and one-day option changes**
   Construct a contract identifier that tracks the same option over time (e.g. using `symbol`). Within each contract, compute one-day changes

   $$\Delta V_t = V_{t+1} - V_t,$$

   and keep only observations for which ΔV_t is defined (i.e. you need both V_t and V_{t+1}).

   Required reporting:
   - summary statistics of V_t and ΔV_t,

- number of contracts retained after constructing ΔV_t.

3. **Step 3: Underlying SPX merge and one-day underlying changes**
Obtain the SPX close series $\{S_t\}$ over the full sample and compute

$$\Delta S_t = S_{t+1} - S_t.$$

Merge S_t and ΔS_t into the option panel by date.

```
import pandas as pd

start = "2023-02-01"
end   = "2023-03-02"

d1 = pd.Timestamp(start).strftime("%Y%m%d")

d2 = pd.Timestamp(end).strftime("%Y%m%d")

url = f"https://stooq.com/q/d/l/?s=^spx&d1={d1}&d2={d2}&i=d"
df = pd.read_csv(url, parse_dates=["Date"]).sort_values("Date")

SPX_price_series = df.set_index("Date")["Close"]
```

Required reporting:

- merge coverage: % of option observations matched to valid $(S_t, \Delta S_t)$,
- summary statistics of S_t and ΔS_t,
- one plot of S_t and one plot of ΔS_t over time.

4. **Step 4: Stripping a daily risk-free curve with the Nelson–Siegel–Svensson (NSS) model and extracting $r_t(\tau)$**
In this step you will construct, for every trading date t, a **smooth risk-free zero-rate curve** and then evaluate it at each option maturity τ in your panel. The output is a **continuously-compounded zero rate** $r_t(\tau)$ that will be used consistently in: (i) discounting $e^{-r_t(\tau)\tau}$ and (ii) the forward mapping $F_t(\tau) = S_t e^{(r_t(\tau)-q)\tau}$.

Data source (US Treasury par yields). Let $y_t(m)$ denote the observed US Treasury **par yield** on date t for a given standard maturity m. We use the Daily Treasury Par Yield Curve archive from the US Treasury website, containing (among others) the tenors:

$$m \in \left\{ \tfrac{1}{12}, \tfrac{2}{12}, \tfrac{3}{12}, \tfrac{6}{12}, 1, 2, 3, 5, 7, 10, 20, 30 \right\} \text{ years.}$$

In practice, yields are provided in annualized % units and must be converted to decimals.

```
import pandas as pd

url = (
 "https://home.treasury.gov/resource-center/data-chart-center/interest-rates/"
 "daily-treasury-rate-archives/par-yield-curve-rates-2020-2023.csv"
)

treasury = pd.read_csv(url)
treasury["date"] = pd.to_datetime(treasury["date"]).dt.normalize()
```

Objective. For each option observation (t, K, T) with year-fraction maturity $\tau = (T - t)/365$, you must attach a risk-free rate

$$r_t(\tau) \quad \text{(continuous compounding)}.$$

To obtain $r_t(\tau)$ you will:

(a) align the Treasury curve to the option trading dates,

(b) fit an NSS parametric yield curve for each date,

(c) evaluate the fitted curve at the option maturity τ,

(d) convert the resulting annual yield quote into a continuously-compounded zero rate.

(a) Date alignment and missing-day handling

Treasury yields are not recorded on weekends and some holidays. Let $\mathcal{T}^{\text{opt}}$ be the set of option trading dates in your panel. Construct a Treasury yield panel indexed by $\mathcal{T}^{\text{opt}}$ by **forward-filling** missing Treasury observations:

$$y_t(m) \;\leftarrow\; y_{\max\{s \le t:\; s \in \mathcal{T}^{\text{tr}}\}}(m), \qquad t \in \mathcal{T}^{\text{opt}},$$

where $\mathcal{T}^{\text{tr}}$ are the dates available in the Treasury file. (Optionally apply backward-fill at the very beginning of the sample if needed.) This ensures that every option date has a full set of yields $\{y_t(m)\}_m$.

(b) Nelson–Siegel–Svensson (NSS) curve

For each date t, we fit a smooth parametric function $y_t(\tau)$ defined for any maturity $\tau > 0$:

$$y_t(\tau) = \beta_{0,t} + \beta_{1,t}\, g_1(\tau; \tau_{1,t}) + \beta_{2,t}\, g_2(\tau; \tau_{1,t}) + \beta_{3,t}\, g_3(\tau; \tau_{2,t}),$$

where the basis functions are

$$g_1(\tau;\lambda) = \frac{1 - e^{-\tau/\lambda}}{\tau/\lambda}, \qquad g_2(\tau;\lambda) = g_1(\tau;\lambda) - e^{-\tau/\lambda}, \qquad g_3(\tau;\lambda) = \frac{1 - e^{-\tau/\lambda}}{\tau/\lambda} - e^{-\tau/\lambda}.$$

Here $(\beta_{0,t}, \beta_{1,t}, \beta_{2,t}, \beta_{3,t})$ control level/slope/curvature, while $(\tau_{1,t}, \tau_{2,t})$ control decay speeds.

(c) Daily calibration via grid search + conditional OLS

On each date t, you observe a discrete set of par yields

$$\{(m_j,\, y_t(m_j))\}_{j=1}^{J}, \qquad J \approx 12.$$

For fixed (τ_1, τ_2) the NSS model is **linear** in the β coefficients. Define the regressor matrix

$$X(\tau_1,\tau_2) = \begin{pmatrix} 1 & g_1(m_1;\tau_1) & g_2(m_1;\tau_1) & g_3(m_1;\tau_2) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & g_1(m_J;\tau_1) & g_2(m_J;\tau_1) & g_3(m_J;\tau_2) \end{pmatrix}, \qquad \mathbf{y}_t = \begin{pmatrix} y_t(m_1) \\ \vdots \\ y_t(m_J) \end{pmatrix}.$$

Then the OLS estimate at (τ_1, τ_2) is

$$\widehat{\boldsymbol{\beta}}_t(\tau_1, \tau_2) = \arg\min_{\boldsymbol{\beta} \in \mathbb{R}^4} \|\mathbf{y}_t - X(\tau_1, \tau_2)\boldsymbol{\beta}\|_2^2 = \left(X^\top X\right)^{-1} X^\top \mathbf{y}_t.$$

We select $(\tau_{1,t}, \tau_{2,t})$ by a grid search:

$$(\widehat{\tau}_{1,t}, \widehat{\tau}_{2,t}) = \arg\min_{(\tau_1,\tau_2)\in\mathcal{G}} \text{SSE}_t(\tau_1, \tau_2), \qquad \text{SSE}_t(\tau_1, \tau_2) := \left\|\mathbf{y}_t - X(\tau_1, \tau_2)\widehat{\boldsymbol{\beta}}_t(\tau_1, \tau_2)\right\|_2^2,$$

with the constraint $\tau_2 \geq \tau_1$ for numerical stability. Finally, define the fitted daily curve

$$\widehat{y}_t(\tau) = y_t\big(\tau; \widehat{\boldsymbol{\beta}}_t, \widehat{\tau}_{1,t}, \widehat{\tau}_{2,t}\big).$$

(d) Evaluating the curve at option maturities

For each option row with maturity τ, compute the annualized fitted yield

$$\widehat{y}_t(\tau) \quad \text{(in decimals, e.g. 0.045 for 4.5\%)}.$$

To avoid the ill-defined $\tau = 0$ limit in the basis functions and in discounting, apply a lower bound:

$$\tau \leftarrow \max\{\tau,\ 1/365\}.$$

(e) Converting Treasury yield quotes to a continuously-compounded zero rate

Treasury par yields are quoted using (approximately) **semiannual compounding**. Therefore the continuously-compounded zero rate is defined by

$$r_t(\tau) = 2\ln\left(1 + \frac{\widehat{y}_t(\tau)}{2}\right).$$

This $r_t(\tau)$ is the rate you must use everywhere in this assignment when pricing/hedging and computing forwards.

Discount factor and forward. Once $r_t(\tau)$ is available for each option observation:

$$P_t(\tau) = e^{-r_t(\tau)\tau}, \qquad F_t(\tau) = S_t\, e^{(r_t(\tau)-q)\tau},$$

where in this lab you may set $q = 0$ unless you explicitly incorporate dividends.

Required reporting. You must report:

- a table/summary of $\{r_t(\tau)\}$ values (min/median/max) across the option panel,
- one plot of the fitted curve $\tau \mapsto r_t(\tau)$ for a representative date (e.g. 2023-02-28),
- verification that every option row has a finite $r_t(\tau)$ and discount factor $P_t(\tau)$.

Remark (interpretation). The fitted object $r_t(\tau)$ is a **zero rate** (continuously-compounded) for maturity τ. In the one-step hedging setting, it is treated as the effective financing rate over the interval $[t, t+\tau]$ used to map spot to forward and to discount payoffs.

5. **Step 4: BS delta hedging and hedging performance [4p]**
This step establishes the baseline hedge performance.

(a) Baseline delta. If a dataset-provided hedge delta is available (e.g. a vendor Black–Scholes delta), use it as baseline:

$$\Delta_t^{\mathrm{BS}} := \delta_t^{\mathrm{vendor}}.$$

If no delta is available, compute a Black–Scholes delta using the market implied volatility σ_{mkt} and a fixed instanteneous interest rate and dividend (r, q). For this you can set dividends equal to zero. If you do not manage to get an interpolated estimate for the risk-free rate as we described in detail in Step 3 above use also a value equal to zero.

(b) Baseline residuals and SSE. Compute residuals

$$\varepsilon_t(\Delta^{\mathrm{BS}}) = \Delta V_t - \Delta_t^{\mathrm{BS}}\,\Delta S_t,$$

and later (after Step 6 filters) compute

$$\mathrm{SSE}(\Delta^{\mathrm{BS}}) = \sum_t \varepsilon_t(\Delta^{\mathrm{BS}})^2.$$

Required reporting:

- summary statistics of $\varepsilon_t(\Delta^{\mathrm{BS}})$,
- baseline $\mathrm{SSE}(\Delta^{\mathrm{BS}})$ after standardized filters (Step 6),
- one plot of baseline delta vs strike for one chosen day and maturity slice.
- Study the misspecification effect on the hedging performance for the implied-volatility parameter. Select a grid of implied volatility parameters and hedge the option with different levels of implied-vol $\Delta_t^{\mathrm{BS}}(\sigma)$. Can you connect your results to the theoretical result you obtained in Exercise 2.

(c) Standardized filters (mandatory). After computing deltas, filter out observations (do not include them) with:

$$\Delta_t^{\mathrm{BS}} \le 0.05, \qquad \Delta_t^{\mathrm{BS}} \ge 0.95, \qquad D_t \le 14.$$

According to Hull, J., and White (2016), filtering out options with extreme delta values and maturity less than 14 days will help you with the analysis of the results.

You must report the remaining row count after each filter.

(e) Standardized bucketing (mandatory). Create:

- 9 moneyness buckets based on Δ^{BS} using equal-width bins over $(0.05, 0.95)$,
- 7 maturity buckets based on D_t using equal-width bins over the filtered maturity range.

This yields $9 \times 7 = 63$ buckets.

(f) Hedging residuals and SSEs. Compute residuals

$$\varepsilon_t(\Delta^{\mathrm{BS}}) = \Delta V_t - \Delta_t^{\mathrm{BS}} \Delta S_t,$$

and compute total SSEs and MSE on the final filtered sample:

$$\mathrm{SSE}(\Delta^{\mathrm{BS}}), \qquad \mathrm{MSE}(\Delta^{\mathrm{BS}}),$$

Required plots:

- Smile plots: $\sigma_{\mathrm{mkt}}(K)$ for one chosen (t, D). You can also construct the smile surfaces $\sigma_{\mathrm{mkt}}(K, t)$ for a choosen day D.
- Delta plots: $K \mapsto \Delta^{\mathrm{BS}}(K)$, for one chosen (t, D).
- Heatmaps of $\mathrm{SSE}_{i,j}(\mathrm{BS})$ and $\mathrm{MSE}_{i,j}(\mathrm{BS})$ across the bucketing you have constructed in Step 5 (e).

6. **Step 7: Standardized outputs for competition (CSV + PDF)**
   To participate in the ranking, you must export:
   (a) `HedgingScoreboard.csv` containing:
   - total SSEs and MSE across the grid we have constructed,
   - bucketed results in tidy long format: one row per bucket (i, j) with SSE and MSE.

   Submission: Upload your notebook, the two CSV files, and a short PDF report containing the required plots and short explanations.

References

- J. Hull and A. White (2016): *Optimal Delta Hedging.*

A Assignment 1

Historical Trend Estimation:

By discretizing (1) along a partition of the interval $[0, T]$ at observation dates $t_0, t_1, \ldots, t_N$, we obtain

$$\frac{S_{t_{k+1}} - S_{t_k}}{S_{t_k}} = \mu(t_{k+1} - t_k) + \sigma(B_{t_{k+1}} - B_{t_k}), \quad k = 0, 1, \ldots, N-1. \tag{7}$$

A natural estimator for the drift parameter μ is given by

$$\hat{\mu}_N := \frac{1}{N} \sum_{k=0}^{N-1} \frac{1}{t_{k+1} - t_k} \left(\frac{S^M_{t_{k+1}} - S^M_{t_k}}{S^M_{t_k}} \right), \tag{8}$$

where $(S^M_{t_{k+1}} - S^M_{t_k})/S^M_{t_k}$, $k = 0, 1, \ldots, N-1$, denotes market returns observed at discrete times $t_0, \ldots, t_N$.

Historical Log-Returns:

Alternatively, by considering log-returns, we have

$$\begin{aligned} \log \frac{S_{t_{k+1}}}{S_{t_k}} &= \log S_{t_{k+1}} - \log S_{t_k} \\ &= \log \left(1 + \frac{S_{t_{k+1}} - S_{t_k}}{S_{t_k}} \right) \\ &\approx \frac{S_{t_{k+1}} - S_{t_k}}{S_{t_k}}. \end{aligned}$$

With uniform time increments $t_{k+1} - t_k = \frac{T}{N}$, we can replace (8) with the telescoping estimator:

$$\frac{1}{N} \sum_{k=0}^{N-1} \frac{1}{t_{k+1} - t_k} \left(\log S_{t_{k+1}} - \log S_{t_k} \right) = \frac{1}{T} \log \frac{S_T}{S_0}. \tag{9}$$

Using (8), the main contribution to the variance of $\hat{\mu}_N$ is

$$\mathrm{Var}(\hat{\mu}_N) = \frac{1}{T^2} \sum_{k=0}^{N-1} \mathrm{Var}(\sigma \Delta B_t) = \frac{\sigma^2}{T}, \tag{10}$$

which implies a standard deviation of $\sigma/\sqrt{T}$. To construct a 95% confidence interval with a 1% window (i.e., $\pm 0.5\%$), we set:

$$q(\alpha) \frac{\sigma}{\sqrt{T}} \leq 0.5\%, \tag{11}$$

leading to $T \geq (1.96\sigma/0.005)^2$. Thus, for a volatility $\sigma = 0.2$ (20%), we require more than 6.146 years to obtain an unbiased drift estimator with a precision of 1%. This considerable timeframe motivates our primary focus on estimating realized volatility rather than drift.

Historical Volatility Estimation:

The volatility parameter σ can be estimated by rearranging (7):

$$\sigma^2 \sum_{k=0}^{N-1} \frac{(B_{t_{k+1}} - B_{t_k})^2}{t_{k+1} - t_k} = \sum_{k=0}^{N-1} \frac{1}{t_{k+1} - t_k} \left(\frac{S_{t_{k+1}} - S_{t_k}}{S_{t_k}} - (t_{k+1} - t_k)\mu \right)^2, \tag{12}$$

leading to the unbiased realized volatility estimator:

$$\hat{\sigma}_N^2 := \frac{1}{N-1} \sum_{k=0}^{N-1} \frac{1}{t_{k+1} - t_k} \left(\frac{S_{t_{k+1}} - S_{t_k}}{S_{t_k}} - (t_{k+1} - t_k)\hat{\mu}_N \right)^2 \tag{13}$$

$$= \frac{1}{N-1} \sum_{k=0}^{N-1} \frac{1}{t_{k+1} - t_k} \left(\frac{S_{t_{k+1}} - S_{t_k}}{S_{t_k}} \right)^2 - \frac{T}{N-1} \hat{\mu}_N^2. \tag{14}$$

The VIX Index:

An alternative approach to estimate market volatility is using the CBOE Volatility Index (VIX), particularly for the S&P 500 Index (SPX). Consider an asset price process S_t satisfying

$$dS_t = rS_t dt + \sigma_t S_t dB_t, \tag{15}$$

which can also be represented as

$$S_t = S_0 \exp\left(\int_0^t \sigma_s dB_s + rt - \frac{1}{2} \int_0^t \sigma_s^2 ds \right), \tag{16}$$

where $(\sigma_t)_{t \geq 0}$ denotes a stochastic volatility process.

Lemma A.1 *Let $\phi \in C^2((0, \infty))$. For all $y > 0$, we have the Taylor-type representation*

$$\phi(x) = \phi(y) + (x - y)\phi'(y) + \int_0^y (z - x)^+ \phi''(z) dz + \int_y^\infty (x - z)^+ \phi''(z) dz \tag{17}$$

for all $x > 0$.

The next proposition shows that the VIX volatility index, defined by

$$\text{VIX}_t := \sqrt{\frac{2e^{r\tau}}{\tau} \left(\int_0^{F_{t,t+\tau}} \frac{P(t, t+\tau, K)}{K^2} dK + \int_{F_{t,t+\tau}}^\infty \frac{C(t, t+\tau, K)}{K^2} dK \right)}, \tag{18}$$

at time $t > 0$, can be interpreted as an average of future volatility. Here, according to CBOE documentation, $\tau = 30$ days, and

$$F_{t,t+\tau} := \mathbb{E}_{\mathbb{Q}}[S_{t+\tau} \mid \mathcal{F}_t] = e^{r\tau} S_t \tag{19}$$

represents the forward price at time $t + \tau$. $P(t, t+\tau, K)$ and $C(t, t+\tau, K)$ denote out-of-the-money put and call option prices at maturity $t + \tau$ with strike K. One can further show that the VIX index at $t \geq 0$ corresponds to the averaged realized variance swap price:

$$\mathrm{VIX}_t = \sqrt{\frac{1}{\tau}\mathbb{E}_{\mathbb{Q}}\left[\int_t^{t+\tau} \sigma_u^2 du \mid \mathcal{F}_t\right]}. \tag{20}$$

The second goal is to estimate the VIX index based on the discretization of (18) and market option prices on the SPX. The strikes for OTM puts and calls are ordered as follows:

$$K_1^{(p)} < \cdots < K_{n_p-1}^{(p)} < K_{n_p}^{(p)} := F_{t,t+\tau} =: K_0^{(c)} < K_1^{(c)} < \cdots < K_{n_c}^{(c)}, \tag{21}$$

and the discretization of (18) becomes:

$$\begin{aligned}
\mathrm{VIX}_t^2 &= \frac{2e^{r\tau}}{\tau}\left(\sum_{i=1}^{n_p-1}\int_{K_i^{(p)}}^{K_{i+1}^{(p)}} \frac{P(t,t+\tau,K_i^{(p)})}{K^2}dK + \sum_{i=1}^{n_c}\int_{K_{i-1}^{(c)}}^{K_i^{(c)}} \frac{C(t,t+\tau,K_i^{(c)})}{K^2}dK\right) \\
&= \frac{2e^{r\tau}}{\tau}\left(\sum_{i=1}^{n_p-1} P(t,t+\tau,K_i^{(p)})\left(\frac{1}{K_i^{(p)}} - \frac{1}{K_{i+1}^{(p)}}\right) + \sum_{i=1}^{n_c} C(t,t+\tau,K_i^{(c)})\left(\frac{1}{K_{i-1}^{(c)}} - \frac{1}{K_i^{(c)}}\right)\right).
\end{aligned}$$